AF
2124
IFW

PTO/SB/21 (12-97)
Approved for use through 9/30/00. OMB 0651-0031
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paper Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/583,747
Filing Date	May 31, 2000
In re Application of:	Harlan SEXTON
Group Art Unit	2124
Examiner Name	Najar, Q.
Attorney Docket Number	50277-0450
Client Docket Number	OID-1997-048-11
Total Number of Pages in This Submission	54

ENCLOSURES ***(check all that apply)***

☒ Fee Transmittal Form
- ☒ Fee Attached

☐ Amendment / Response
- ☐ After Final
- ☐ Affidavits/declaration(s)

☐ Extension of Time Request

☐ Express Abandonment Request

☐ Information Disclosure Statement

☐ Certified Copy of Priority Document(s)

☐ Response to Missing Parts/ Incomplete Application
- ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53

☐ Assignment Papers *(for an Application)*

☐ Drawing(s)

☐ Licensing-related Papers

☐ Petition Routing Slip (PTO/SB/69) and Accompanying Petition

☐ To Convert a Provisional Application

☐ Power of Attorney, Revocation Change of Correspondence Address

☐ Terminal Disclaimer

☐ Small Entity Statement

☐ Request of Refund

☐ After Allowance Communication to Group

☐ Appeal Communication to Board of Appeals and Interferences

☒ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*

☐ Proprietary Information

☐ Status Letter

☐ Additional Enclosure(s) *(please identify below)*:

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm *or* Individual name	DITTHAVONG & CARLSON, P.C. Stephen C. Carlson, Reg. No. 39929
Signature	
Date	September 7, 2004

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Assistant Commissioner for Patents, Alexandria, VA 22313-1450 on this date: 9/7/04

Type or printed name	Linda V. Wiley		
Signature		Date	September 7, 2004

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time your are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

PTO/SB/17 (01-03)
Approved for use through 04/30/2003. OMB 0651-0032
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL for FY 2004

Effective 10/01/2003. Patent fees are subject to annual revision.

□ Applicant Claims small entity status. See 37 CFR 1.27

TOTAL AMOUNT OF PAYMENT ($) 330.00

Complete if Known	
Application Number	09/583,747
Filing Date	May 31, 2000
First Named Inventor	Sexton
Examiner Name	Najar, Q.
Art Unit	2124
Attorney Docket No.	50277-0450

METHOD OF PAYMENT *(check all that apply)*

□ Check ☒ Credit card □ Money Order □ Other □ None

□ Deposit Account

Deposit Account Number

Deposit Account Name

The Commissioner is authorized to: *(check all that apply)*

☒ Charge fee(s) indicated below □ Credit any overpayments

□ Charge any additional fee(s) during the pendency of this application

□ Charge fee(s) indicated below, **except for the filing fee**

to the above-identified deposit account.

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1001	770	2001	385	Utility filing fee	
1002	340	2002	170	Design filing fee	
1003	530	2003	265	Plant filing fee	
1004	770	2004	385	Reissue filing fee	
1005	160	2005	80	Provisional filing fee	

SUBTOTAL (1) ($)

2. EXTRA CLAIM FEES FOR UTILITY AND REISSUE

			Extra Claims		Fee from below		Fee Paid
Total Claims	22	-22**=	0	x	0	=	0
Independent Claims	4	- 4**=	0	x	0	=	0
Multiple Dependent					0	=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
1202	18	2202	9	Claims in excess of 20
1201	86	2201	43	Independent claims in excess of 3
1203	290	2203	145	Multiple dependent claim, if not paid
1204	86	2204	43	**Reissue independent claims over original patent
1205	18	2205	9	**Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 0.00

*** or number previously paid, if greater; For Reissues, see above*

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1051	130	2051	65	Surcharge – late filing fee or oath	
1052	50	2052	25	Surcharge – late provisional filing fee or cover sheet	
1053	130	1053		Non-English specification	
1812	2,520	1812		For filing a request for *ex parte* reexamination	
1804	920*	1804		Requesting publication of SIR prior to Examiner action	
1805	1,840*	1805		Requesting publication of SIR after Examiner action	
1251	110	2251	55	Extension for reply within first month	
1252	420	2252	210	Extension for reply within second month	
1253	950	2253	475	Extension for reply within third month	
1254	1,480	2254	740	Extension for reply within fourth month	
1255	2,010	2255	1,005	Extension for reply within fifth month	
1401	330	2401	165	Notice of Appeal	
1402	330	2402	165	Filing a brief in support of an appeal	330.00
1403	290	2403	145	Request for oral hearing	
1451	1,510	1451		Petition to institute a public use proceeding	
1452	110	2452	55	Petition to revive – unavoidable	
1453	1,330	2453	665	Petition to revive – unintentional	
1501	1,330	2501	665	Utility issue fee (or reissue)	
1502	480	2502	240	Design issue fee	
1503	640	2503	320	Plant issue fee	
1460	130	1460		Petitions to the Commissioner	
1807	50	1807		Processing fee under 37 CFR 1.17(q)	
1806	180	1806		Submission of Information Disclosure Stmt	
8021	40	8021	40	Recording each patent assignment per property (times number of properties)	
1809	770	2809	385	Filing a submission after final rejection (37 CFR § 1.129(a))	
1810	770	2810	385	For each additional invention to be examined (37 CFR § 1.129(b))	
1801	770	2801	385	Request for Continued Examination (RCE)	
1802	900	1802		Request for expedited examination of a design application	

Other fee (specify) ____________________

*Reduced by Basic Filing Fee Paid

SUBTOTAL (3) ($) 330.00

SUBMITTED BY				Complete *(if applicable)*	
Name (Print/Type)	Stephen C. Carlson	*Registration No. (Attorney/Agent)*	39929	*Telephone*	703-425-8516
Signature				*Date*	September 7, 2004

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

This collection of information is required by 37 CFR 1.17 and 1.27. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 37 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, Washington, DC 20231.**

If you need assistance in completing the form, call 1-800-PTO-9199 (1-800-786-9199) and select option 2.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:			
	Harlan SEXTON	Conf. No.:	4124
Application No.:	09/583,747	Group Art Unit:	2124
Filed:	May 31, 2000	Examiner:	Najar, Q.
Attorney Docket:	50277-0450		
Client Docket:	OID-1997-048-11		

For: DIAGNOSTIC METHOD AND ARTICLE FOR IDENTIFYING SIGNIFICANT EVENTS

APPEAL BRIEF

Honorable Commissioner for Patents
Alexandria, VA 22313-1450

Dear Sir:

This Appeal Brief is submitted, in triplicate, in support of the Notice of Appeal dated July 6, 2004.

I. REAL PARTY IN INTEREST

Oracle International Corporation is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals and interferences.

III. STATUS OF THE CLAIMS

Claims 1-22 are pending in this appeal. No claim is allowed. This appeal is therefore taken from the final rejection of claims 1-22 on May 5, 2004.

IV. STATUS OF AMENDMENTS

No amendment to the claims has been filed since the final rejection of the claims on May 5, 2004.

V. SUMMARY OF THE INVENTION

The present invention addresses problems associated with managing memory in a dynamic run-time environment by identifying events of significance during execution of a program, for example, by identifying objects which are migrated from one memory, such as a shorter-duration memory (e.g., a call memory), into another memory, such as a longer duration memory (e.g., a session memory) (Specification, page 6, lines 1-3; page 7, lines 5-11). The present invention is directed to logging backtraces in a log file, and additionally including tags during execution of a program with information that can categorize the backtraces. Further, certain tags can also be marked as "interesting" in the log file during execution of the program. A report is generated from the log file, showing one or more of the backtraces associated with the interesting tags. Consequently, significant events can be automatically identified from a set of loggable events when the significance of the events can only be determined after the logging of the event occurs. (Specification, page 9, lines 3-9)

In one embodiment, backtraces are logged whenever a memory management routine to allocate memory for an object is called. These backtraces are also tagged with the starting address of allocated memory. When the objects are migrated at the end of a call, their starting address is marked in the backtrace log file as "interesting." Consequently, the generated report will show the backtraces associated with the allocation of objects that were later migrated. Thus, backtraces of migrated objects are produced when the objects were allocated, even though it can only be determined later that a particular allocated object was later migrated to session memory (Specification, page 9, lines 10-19).

Accordingly, one aspect of the invention relates to a method and software for analyzing a program, in which stack traces and tags are logged in a log file at points during execution of the program, and one or more tags (e.g., "interesting" tags) are recorded within the log file as one or more marked tags (Specification, page 9, lines 20-24).

Another aspect pertains to a method and software for producing a diagnostic report for a program, which includes accessing a log file comprising stack traces and associated tags logged at points during execution of the program as well as one or more marked tags (e.g., "interesting" tags). The diagnostic report is produced based on the log file (Specification, page 9, line 25 - page 10, line 2).

VI. <u>ISSUES</u>

A. Whether claims 1-18 are anticipated under 35 U.S.C § 102 by *Arsenault* (US 5,408,650).

B. Whether claims 19-22 are obvious under 35 U.S.C. § 103 based on *Arsenault* in view of *Elliott et al.* (US 4,945,474).

VII. GROUPING OF CLAIMS

The claims should not be regarded as all standing together since the claims recite respective limitations that render each of the claims separately patentable. For the purposes of this appeal, the following groups are recognized:

A. Claims 1-3, 6-7, 10-12, and 15-16.

B. Claims 4, 8, 13, and 17.

C. Claims 5, 9, 14, and 18.

D. Claims 19-22.

VIII. ARGUMENT

A. CLAIMS 5, 9, 14, AND 18 ARE NOT ANTICIPATED BECAUSE *ARSENAULT* FAILS TO DISCLOSE "MIGRATED OBJECTS."

To anticipate a patent claim, every element and limitation of the claimed invention must be found in a single prior art reference, arranged as in the claim. *Karsten Mfg. Corp. v. Cleveland Golf Co.*, 242 F.3d 1376, 1383, 58 USPQ2d 1286, 1291 (Fed. Cir. 2001); *Scripps Clinic & Research Foundation v. Genentech, Inc.*, 927 F.2d 1565, 1576, 18 USPQ2d 1001, 1010 (Fed. Cir. 1991).

The rejection of claims 5, 9, 14, and 18 over *Arsenault* is improper because the applied reference does not disclose the limitations of the claims. For example, claims 5, 9, 14, and 18 recite:

> 5. (Previously Presented) The method according to claim 1, wherein:
> the tags indicate respective addresses of allocated objects; and
> the one or more marked tags indicate one or more respective addresses of migrated objects.
>
> 9. (Previously Presented) The method according to claim 6, wherein:
> the tags indicate respective addresses of allocated objects; and

the one or more marked tags indicate one or more respective addresses
of migrated objects.

Claims 14 and 18 are computer-readable medium claims corresponding to claims 5 and 9, respectively.

Arsenault does not have any disclosure of "migrated objects" as recited in claims 5, 9, 14, and 18. The Examiner's response on page 10 of the Office Action states that "deallocation routine indicates migrated objects; that is, objects that have been deallocated are migrated objects. Allocated objects that have been deallocated are objects that have migrated from their current memory locations. Therefore, the one or more marked tags indicate one or more respective addresses of migrated objects, that is, addresses of allocated objects that have been migrated." However, *Arsenault* defines the term "deallocate" as a "release" of memory locations. For example, at col. 1: 27-44, *Arsenault* states (emphasis added):

> When the program completes its manipulation of various data from the data base, the program can then **deallocate, or release, the dynamically allocated memory locations** used to hold this data. **This frees the locations for other uses and/or for re-use by the same program**. Similarly, **if various results of the data manipulation are no longer needed, the program can release the memory locations used to store the results**. Such allocations and **releases** may occur many times throughout the execution of the program.
>
> To allocate and **release** memory locations, the programs invoke memory allocation and **deallocation** routines, which are low-level operating system routines, commonly referred to as "service routines," that control the actual allocation and **release** of the system memory locations. There are various types of memory allocation/**deallocation** routines directed to particular arrangements of memory locations.

Similarly, at least at col. 2: 4-36, col. 2: 46-51, col. 3: 48-61, col. 5: 34-35, and col. 6: 35, *Arsenault* clearly indicates that "deallocate" or "deallocation" refers to a "release" of memory locations. Nowhere does *Arsenault* suggest or disclose any form of "migrated objects," which, at best, may or may not involve a "deallocation" when a copy of "migrated" subject matter is stored at another memory location.

Moreover, claims 5, 9, 14, and 18 recite that the "the one or more **marked tags indicate** one or more respective **addresses** of **migrated objects**" (emphasis added). Unless the patent otherwise provides, a claim term cannot be given a different meaning in the various claims of the same patent. In the rejection of parent claim 1, the "marked tag" is read on a creation count stored in a call-stack, an ordinal number indicating a number of blocks allocated to the program (col. 6:41-49; Fig. 2, item 27). The Office Action (p. 3) further includes "the address '00126398' of the location at the beginning of the segment; the number of locations '00001410' in the segment; the address '001277A8' of the location at the end of the segment," which does not indicate an address of a "migrated object," and then contends that "'creation count' is interpreted as 'marked tags'". The cited col. 6:41–49 discusses contents of the call-stack and cols. 6:50 - col. 7:4 further discusses the contents as including addresses of memory locations which contains code of portions of routines and names of related modules. Thus, there is no teaching of marked tags indicating "one or more respective **addresses** of **migrated objects**," as recited in claims 5, 9, 14, and 18.

B. *ARSENAULT* FAILS TO ANTICIPATE CLAIMS 1-18 BECAUSE *ARSENAULT* DISCLOSES NEITHER "LOGGING A PLURALITY OF STACK TRACES ... IN A LOG FILE" NOR "ACCESSING A LOG FILE COMPRISING A LIST OF STACK TRACES."

Turning now to the rejection of claims 1-18, this rejection is respectfully traversed because *Arsenault* does not disclose the limitations recited in independent claims 1, 6, 10, and 15. For example, independent claims 1 and 10 recite "logging a plurality of stack traces and respective tags in a log file," and independent claims 6 and 15 recite "accessing a log file comprising a list of stack traces and respective tags."

Arsenault contains no explicit disclosure of a "log file" and at best discloses a representation displayed to the user on the screen of a display device that includes "a listing **26** of the call-stack associated with a selected memory segment" (col. 6:2-4, note singular "call-stack") which has been produced by a memory analysis system using information in symbol tables formulated by a debugger **18** (col. 5:37-51). Specifically, *Arsenault* discloses a graphic representation of a map of allocated memory segments depicted by segment type and various listings shown on a display device to a user (cols. 5:65–6:4), but not the "recording within the log file one or more of the tags as one or more marked tags" as recited in independent claims 1 and 10 and "accessing a log file comprising a list of stack traces and respective tags."

In response to Appellant's arguments in the response filed March 17, 2004, the Examiner, p. 13, resorted to inherency to fill out the disclosure missing in *Arsenault*: "Arsenault inherently teaches a log file because there is a debugger, which stores the information it collects in a log file, see column 3, lines 37-45." However, col. 3:37-45 has nothing to do with a log file, but with accessing the "'symbols' and locations of the **compiled executable version** of the program" (col. 3: 40-41, emphasis added). A compiled executable version of a program is not a log file,[1] and, even if it were, debuggers do not write stack traces and other debugging output to compiled executable versions of programs. Further, the content of the "information" referred to in col. 3:37-45 is discussed in more depth in col. 5:40-53, where the "information" is a "symbol table;" however, a symbol table does not include a stack trace.

[1] Appellant's representatives apologize for a typographical error in the response dated March 17, 2004 (p.9: 9) which confusingly read "A compiled executable version of a program is not a program."

Moreover, the rejection does not meet the standard imposed by the Federal Circuit in using inherency. Although "inherency places subject matter in the public domain as well as express disclosure," *Schering Corp. v. Geneva Pharms., Inc.*, No. 02-1540 (Fed. Cir., August 1, 2003), slip op. at 9, it must be clear that the missing descriptive matter is necessarily present in the reference to establish inherency. *In re Roberston*, 49 USPQ2d 1949, 1951 (Fed. Cir., 1999). Under the principles of inherency, the prior art must necessarily function in accordance with, or include, the claim limitations. *MEHL/Biophile Int'l.*, 52 USPQ2d 1303 (Fed. Cir. 1999); see also *Schering Corp.*, *id.* at 8 ("necessarily and inevitably"). As explained in *MEHL/Biophile Int'l*:

> Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient. If, however, the disclosure is sufficient to show that the natural result flowing from the operation as taught would result in the performance of the questioned function, it seems to be well settled that the disclosure should be regarded as sufficient.

The attempt to base the rejection on something other than what *Arsenault* teaches falls far short of these requirements for inherency. For example, there is no evidence on the record that "there is a debugger, which stores the information it collects in a log file." Moreover, it is not a necessary requirement for debuggers to have log files, especially on microprocessor systems. If this statement is "well-known," then it is traversed and the Patent Office is requested to articulate and place on the record the "common knowledge" used to negate patentability. *In re Zurko*, 258 F.3d 1379, 1386, 59 USPQ2d 1693, 1697 (Fed. Cir. 2001); *In re Sang Su Lee*, No. 00-1158 (Fed. Cir., Jan. 18, 2002). As another example, even if some evidence were to be shown of a debugger using a log file, one must still show a debugger that logs stack traces in particular.

The Examiner, page 14, further asserts:

> Debuggers write stack traces and other debugging output to a *log file*. That is, debuggers need to store the information it collects in a *log file* for later

> processing. Arsenault teaches that "the memory analysis system communicates with a debugger. The debugger analyzes the execution of the program in a conventional manner and relates 'symbols' and locations of the compiled execution version of the program, that is, the image file, to symbols and locations in the source code version of the program. The memory analysis system then uses this ***information*** to generate the call-stacks and make available the related lines of source code."; see column 3, lines 37-45 (emphasis added). Therefore, the debugger stores this information in a *log file* for later processing.

As best understood, the Examiner is asserting that because the memory analysis system of *Arsenault* uses "information" from the image file to generate a call stack, *Arsenault* must have a log file containing stack traces. This is fallacious. As discussed previously, the content of the "information" referred to in col. 3:37-45 is not a stack trace but a "symbol table." *Arsenault* merely discloses that the memory analysis system **12** produces the associated call-stacks using the "information" of the **symbol tables** formulated by the debugger **18** to translate the addresses in a stack trace. There is no mention whatever of any use of a "log file" to store a stack trace, either by the debugger **18** or by the memory analysis system **12**.

C. *ARSENAULT* FAILS TO ANTICIPATE CLAIMS 4, 8, 13, AND 17 BECAUSE *ARSENAULT* FAILS TO DISCLOSE "IDENTIFYING A LAST STACK TRACE THAT IS ASSOCIATED WITH ONE OF THE ONE OR MORE MARKED TAGS; AND PRODUCING THE REPORT BASED ON THE IDENTIFIED LAST STACK TRACE."

With regard to the rejection of claims 4, 8, 13, and 17, this rejection is respectfully traversed because *Arsenault* does not disclose the limitations recited in dependent claims 4, 8, 13, and 17. For example, claims 4, 8, 13, and 17 recite "identifying a last stack trace that is associated with one of the one or more marked tags; and producing the report based on the identified last stack trace."

Regarding claim 4, the Examiner contends, "Arsenault further discloses identifying the last stack trace associated with any of the one or more marked tags; and producing said report based on the identified stack traces," citing col. 3:11-22 and col. 11:8-12, and then states, "For a finite quantity of identified stack traces, the feature is inherent in Arsenault's system to enable specific reports to be generated for any of the identified stack traces (col. 12, li. 6-8)." (Office Action, page 4) However, the portions of *Arsenault* cited by the Examiner refer to the memory analysis system evaluating memory allocation or deallocation routines associated with memory events initiated by the application program and then updating "the display appropriately by adding or removing information" (col. 3:20-22), processing a user-initiated request such as information about the accessibility of a particular memory location (col. 10:57-65) and sending updated instructions to the display unit after processing all messages from the display unit (col. 11:8-12), and sending to the display unit information that associates a segment type with a call-stack and with lines of source code (col. 12:6-8). Nowhere is there any mention of "identifying a last stack trace that is associated with one of the one or more marked tags" as recited by each of claims 4, 8, 13, and 17, and the Examiner does not contend that this recited feature is taught by *Arsenault*. The Examiner instead apparently relies on a statement, "the feature is inherent in Arsenault's system" to provide a teaching of the recited features.

This rejection too does not meet the standard imposed by the Federal Circuit in using inherency. *Schering Corp. v. Geneva Pharms., Inc.*, No. 02-1540 (Fed. Cir., August 1, 2003), slip op. at 9. *In re Roberston*, 49 USPQ2d 1949, 1951 (Fed. Cir., 1999). *MEHL/Biophile Int'l.*, 52 USPQ2d 1303 (Fed. Cir. 1999); see also *Schering Corp.*, *id.* at 8 ("necessarily and inevitably").

No stack trace in *Arsenault* is identified as "last," nor does such an identification seem necessary or useful in the *Arsenault* system. Indeed, if the program being debugged by *Arsenault* is caught in an infinite loop, there may never be a "last" stack trace, at least in any relevant sense. Thus, *Arsenault*'s use of a display of information related to memory does not inherently or otherwise disclose "enable specific reports to be generated for any of the identified stack traces" as contended by the Examiner, much less the recited "producing the report based on the identified last stack trace."

D. THERE IS NO MOTIVATION TO COMBINE *ARSENAULT* AND *ELLIOT ET AL.* FOR CLAIMS 19-22.

Obviousness rejections require some evidence in the prior art of a teaching, motivation, or suggestion to combine and modify the prior art references. See, e.g., *McGinley v. Franklin Sports, Inc.*, 262 F.3d 1339, 1351-52, 60 USPQ2d 1001, 1008 (Fed. Cir. 2001); *Brown & Williamson Tobacco Corp. v. Philip Morris Inc.*, 229 F.3d 1120, 1124-25, 56 USPQ2d 1456, 1459 (Fed. Cir. 2000); *In re Dembiczak*, 175 F.3d 994, 999, 50 USPQ2d 1614, 1617 (Fed. Cir. 1999).

The obviousness rejection of claims 19-22 is respectfully traversed because there is no motivation to combine *Arsenault* and *Elliott et al.* As best understood, the Examiner attempts to read the recited "log file" on an "inherent" log file in debuggers. However, as explained above, this use of inherency is insufficient as a matter of law. Even if the Examiner were to find a reference explicitly disclosing this, there is still no motivation for one of ordinary skill in the art to modify the supposed debugger log files that log output with the non-analogous *Elliott et al.* Indeed, *Elliott et al.* is not directed to processing supposed debugger log files at all but to processing relational database recovery logs. The formats are different, and all the special processing in *Elliott et al.* to handle I/O errors and system crashes when logging database transactions is irrelevant to *Arsenault*, which does not use database transactions. The proposed modification of using relational database recovery logs would change *Arsenault*'s principle of operation. If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959). MPEP § 2143.01

Moreover, *Elliott et al.* is not analogous prior art. "In order to rely on a reference as a basis for rejection of an applicant's invention, the reference must either be in the field of the applicant's endeavor or, if not, then be reasonably pertinent to the particular problem with which the inventor was concerned." *In re Oetiker*, 977 F.2d 1443 (Fed. Cir. 1992); see also *In re Clay*, 966 F.2d 656 (Fed. Cir. 1992). *Elliott et al.* is directed to processing relational database recovery logs to restore a database after I/O error (Abstract), and does not involve "analyzing a program" or "producing a diagnostic report for a program" as recited by claims 19-22.

IX. CONCLUSION AND PRAYER FOR RELIEF

Appellants, therefore, request the Honorable Board to reverse each of the Examiner's rejections.

Respectfully Submitted,

DITTHAVONG & CARLSON, P.C.

September 7, 2004
Date

Margo Livesay
Margo Livesay, Ph.D.
Reg. No. 41,946

Stephen C. Carlson
Reg. No. 39,929
Attorneys for Applicant(s)

10507 Braddock Rd, Suite A
Fairfax, VA 22032
Tel. 703-425-8501
Fax. 703-425-8518

APPENDIX

1. (Previously Presented) A method for analyzing a program, comprising the steps of:

logging a plurality of stack traces and respective tags in a log file at respective points during execution of the program; and

recording within the log file one or more of the tags as one or more marked tags.

2. (Original) The method according to claim 1, further comprising the step of:

producing a report based on the log file.

3. (Previously Presented) The method according to claim 2, wherein the step of producing the report includes:

identifying one or more of the stack traces that are associated with any of the one or more tags marked; and

producing the report based on the identified one or more of the stack traces.

4. (Previously Presented) The method according to claim 2, wherein producing the report includes:

identifying a last stack trace that is associated with one of the one or more marked tags; and

producing the report based on the identified last stack trace.

5. (Previously Presented) The method according to claim 1, wherein:

the tags indicate respective addresses of allocated objects; and

the one or more marked tags indicate one or more respective addresses of migrated objects.

6. (Previously Presented) A method for producing a diagnostic report for a program, comprising the steps of:

accessing a log file comprising a list of stack traces and respective tags at associated points during execution of the program and comprising one or more marked tags; and

producing the diagnostic report based on the log file.

7. (Previously Presented) The method according to claim 6, wherein the step of producing the report includes:

identifying one or more of the stack traces that are associated with any of the one or more marked tags; and

producing the report based on the identified one or more of the stack traces.

8. (Previously Presented) The method according to claim 6, wherein producing the report includes:

identifying a last stack trace that is associated with one of the one or more marked tags; and

producing the report based on the identified last stack trace.

9. (Previously Presented) The method according to claim 6, wherein:

the tags indicate respective addresses of allocated objects; and

the one or more marked tags indicate one or more respective addresses of migrated objects.

10. (Previously Presented) A computer-readable medium bearing instructions for analyzing a program, said instructions being arranged to cause one or more processors upon execution thereby to perform the steps of:

logging a plurality of stack traces and respective tags in a log file at respective points during execution of the program; and recording within the log file one or more of the tags as one or more marked tags.

11. (Original) The computer-readable medium according to claim 10, further bearing instructions for performing the step of:

producing a report based on the log file.

12. (Previously Presented) The computer-readable medium according to claim 11, wherein the step of producing the report includes:

identifying one or more of the stack traces that are associated with any of the one or more marked tags; and

producing the report based on the identified one or more of the stack traces.

13. (Previously Presented) The computer-readable medium according to claim 11, wherein producing the report includes:

identifying a last stack trace that is associated with one of the one or more marked tags; and

producing the report based on the identified last stack trace.

14. (Previously Presented) The computer-readable medium according to claim 10, wherein:

the tags indicate respective addresses of allocated objects; and

the one or more marked tags indicate one or more respective addresses of migrated objects.

15. (Previously Presented) A computer-readable medium bearing instructions for producing a diagnostic report for a program, said instructions being arranged to cause one or more processors upon execution thereby to perform the steps of:

accessing a log file comprising a list of stack traces and respective tags at associated points during execution of the program and comprising one or more marked tags; and producing the diagnostic report based on the log file.

16. (Previously Presented) The computer-readable medium according to claim 15, wherein the step of producing the report includes:

identifying one or more of the stack traces that are associated with any of the one or more marked tags; and

producing the report based on the identified one or more of the stack traces.

17. (Previously Presented) The computer-readable medium according to claim 15, wherein producing the report includes:

identifying a last stack trace that is associated with one of the one or more marked tags; and

producing the report based on the identified last stack trace.

18. (Previously Presented) The computer-readable medium according to claim 15, wherein:

the tags indicate respective addresses of allocated objects; and

the one or more marked tags indicate one or more respective addresses of migrated objects.

19. (Previously Presented) The method according to claim 4, wherein the step of producing the report includes:

processing the log file from the end backward until the beginning.

20. (Previously Presented) The method according to claim 8, wherein the step of producing the report includes:

processing the log file from the end backward until the beginning.

21. (Previously Presented) The computer-readable medium according to claim 13, wherein the step of producing the report includes:

processing the log file from the end backward until the beginning.

22. (Previously Presented) The computer-readable medium according to claim 17, wherein the step of producing the report include:

processing the log file from the end backward until the beginning.